SPECIFICATION

TEMPLATE SYSTEM, SOFTWARE PROGRAM USING THE SAME TEMPLATE SYSTEM, AND METHOD USING THE SAME SOFTWARE PROGRAM

Technical Field

[0001] The present invention relates to a template system that can be used suitably, for example, as a method for dynamically preparing a Web page or a method for generating a file or data for exchanging information. More specifically, the present invention relates to a template system in which a presentation layer is separated into presentation data and presentation logic, while a change portion of the presentation data is abstracted into a start part, an intermediate part, and an end part.

Background Art

[0002] In a software program that performs display or output of any kind, its internal description can conceptually be separated into a 'business logic layer (or simply a logic layer)' that manages 'what' should be displayed and a 'presentation layer' that manages 'how' the display should be performed.

[0003] When a 'program for displaying a list of

employees' is assumed, the business logic layer determines 'what' should be displayed (for example, the names, the employee numbers, and the extension numbers), and then prepares the data. Then, the presentation layer determines 'how' the prepared data should be displayed (such as the display format, the character size, and the color), and then displays the data.

[0004] An example of means for separating the business logic layer and the presentation layer from each other is a 'template system.' In the 'template system,' data in a format to be displayed is prepared as a template (a pattern), then read by a main program, and then outputted after a necessary portion is re-written.

[0005] That is, in the template system, the roles are divided such that the main program takes charge of the business logic layer (what should be displayed) while the template takes charge of the presentation layer (how the display should be performed), so that the business logic layer and the presentation layer are separated from each other.

[0006] Advantages are obtained when the business logic layer and the presentation layer are separated from each other by using the above-described template system. These advantages include: that various display formats can be

supported merely by changing solely the presentation layer (that is, the template) (at this time, the business logic layer (the main program) need not to be changed); and that since a change in one part does not affect the other part, that is, since a change can be made independently of each other, the roles can easily be divided between a programmer who takes charge of the main program and a designer who takes charge of the template.

[0007] Because of these advantages, the template system is employed especially in Web applications where changes in the display screen are frequently generated.

Non-Patent Document 1: Velocity, http://jakarta.apache.org/velocity/

Non-Patent Document 2: XMLC, http://xmlc.enhydra.org/

Disclosure of the Invention

Problems to be Solved by the Invention

[0008] Here, the above-described presentation layer (that is, the template) can further be divided into 'presentation data (data necessary for display, such as an HTML tag, the character size, the color, and the table frame width)' and presentation logic (logic necessary for display (such as a repeat and a conditional branch)). For example, when a list of employees is displayed in a table format, the

table size, the frame width and color, etc., are recognized as presentation data, while the operation of 'repeating' the display by the number of employees is recognized as presentation logic.

[0009] Here, it should be noted that the 'presentation logic' definitely indicates 'logic necessary for display,' and hence is obviously different from the above-described business logic the aim of which is to prepare data, although these belong to the same concept of logic. Thus, the presentation logic cannot be included in the business logic layer.

[0010] In a conventional template system, the aim has been to separate a business logic layer (a main program) and a presentation layer (a template) from each other. Thus, both of 'presentation data' and 'presentation logic' have been mixed and have never been separated within a template serving as the presentation layer. Fig. 15 is a diagram showing an example of the prior art, and shows a state that both of 'presentation data' and 'presentation logic' are mixed within a presentation layer that constitutes a template.

[0011] As shown in this prior art, in a configuration where 'presentation data' and 'presentation logic' are not separated from each other in a presentation layer, the

following problems can arise. A first problem is that when one of these is changed, the other part can also be changed by mistake with a high probability. A second problem is that the presentation data or the presentation logic cannot independently be extracted and reused. A third problem is that since a program designer is, in general, not very skilled in describing logic, the designer who takes charge of the template could not handle the presentation logic when the logic becomes complicated.

[0012] Thus, a main object of the present invention is to provide a template system that can solve all problems described above.

Means for Solving the Problem

[0013] The present invention provides a template system including a business logic layer and a presentation layer, wherein the presentation layer is separated into presentation data and presentation logic, while a change portion of the presentation data is abstracted into a start part, an intermediate part, and an end part. Further, the present invention provides a template system wherein the presentation logic adopts, as an operation target, each or all of the start part, the intermediate part, and the end part in the presentation data.

[0014] Further, the present invention provides: a software program at least including the above-described template system inside the program; a method for generating a Web page by using the software program; and a method for generating a file or data for exchanging information by using the software program. In the case that a Web page is generated or that a file or data for exchanging information is generated, presentation data or presentation logic is frequently changed. Thus, the software program according to the present invention is suitably applied.

Effects of the Invention

[0015] According to the present invention, since presentation data and presentation logic are completely separated from each other in a presentation layer, even when one of these is changed, the other part is not affected. Thus, an effect is obtained that the presentation data and the presentation logic can be independently changed and that changing the other part by mistake is avoided.

[0016] Further, any one of the presentation data and the presentation logic can be independently extracted and reused. This permits support for any display format and output format.

Best Mode for Carrying Out the Invention

[0017] Preferred embodiments of a template system

according to the present invention are described below with reference to the accompanying drawings. First, Fig. 1 is a diagram showing a basic concept of a template system according to the present invention.

[0018] As shown in Fig. 1, in the template system according to the present invention, a presentation layer P serving as a template is divided into two, so that presentation data D and presentation logic L are independently described.

[0019] The presentation data D shown in Fig. 1 is a description concerning data necessary for display, such as an HTML tag, the character size, the color, and the table frame width, while the presentation logic L is a description concerning logic necessary for display such as a repeat and a conditional branch.

[0020] Then, in the present invention, in the presentation data D, a 'mark' of any kind is attached to a portion to be changed (hereinafter, referred to as a 'change portion').

[0021] For example, in Fig. 1, 'id="users"' serves as the mark. Then, the change portion where the mark is attached in the presentation data D is abstracted and separated into start part/intermediate part/end part (see presentation data D in Fig. 1).

[0022] Here, the configuration in which the change portion is abstracted into start part/intermediate part/end part only requires that abstracted elements serving as a start part, an intermediate part, and an end part should be present in the presentation data D regardless of the format. Thus, the configuration includes: for example, a concept that three abstracted elements each serving as a start part, an intermediate part, and an end part are treated; and a concept (described later) that two abstracted elements each serving as paired 'start part/end part' and an 'intermediate part' are treated.

[0023] Next, in the presentation logic L shown in Fig. 1, operations such as a repeat and a branching are performed on the change portion where the mark is attached in the presentation data D. These operations are devised so as to be capable of being performed on each or all of start part/intermediate part/end part.

[0024] According to this device, in the presentation logic L, when a change portion in the presentation data D is abstracted into start part/intermediate part/end part, an advantage is obtained, for example, that an arbitrary operation is allowed such that the intermediate part is solely repeated or alternatively the entirety of start

part/intermediate part/end part is repeated. Here, in the configuration of the presentation logic L shown in Fig. 1, the intermediate part is solely repeated.

[0025] Here, employable methods for attaching a mark to the change portion in the presentation data D include: (a) a method for embedding a symbol or a keyword serving as a mark into the presentation data D (for example, 'id="users"' in Fig. 1 is this method); (2) a method for considering a character string or a pattern that appears in the presentation data D as a mark; and (3) a method for considering the number of characters or bytes counted from the beginning or the end of the presentation data D as a mark.

[0026] According to these methods, description of the presentation logic L is never mixed into the presentation data D. On the other hand, description concerning the presentation data D is never mixed into the presentation logic L. That is, the presentation data D and the presentation logic L can be completely separated from each other.

[0027] Further, these methods can be applied regardless of the format of the presentation data D. That is, the present invention has an advantage capable of supporting any format of output and display.

[0028] Embodiments of the present invention are

described below in further detail with reference to specific examples shown in Fig. 2 and the subsequent drawings. In these examples, Web applications for outputting an HTML file are assumed. In these cases, the presentation data D is in an HTML format. Here, the presentation data D in the scope of the present invention is not limited to the HTML format.

[0029] First, presentation data D as shown in Fig. 2 is prepared. Then, a mark is attached to a change portion (see Fig. 2). In this example, two kinds of marks are used. More specifically, in Fig. 2, 'id="users"' designated by a symbol D_1 indicates that the part including the line between HTML tags '
' and '
' should be considered as a change portion. Further, '${user}' designated by a symbol D_2 in Fig. 2 indicates that the line should be replaced by the value of a variable 'user.'

[0030] From the presentation data D having this configuration, four macros are automatically generated on the basis of the above-described mark (see Fig. 3). In this example, the following macros are automatically generated.

[0031] A macro 'head_users' designated by a symbol M_1 in Fig. 3 indicates a start part (<ul>). Further, in this Fig. 3, a macro 'body_users' designated by a symbol M_2 indicates an intermediate part, that is, the part (
#{user})

between the start part and the end part. A macro 'foot_users' designated by a symbol M_3 in Fig. 3 indicates an end part (). Further, a macro 'elem_users' designated by a symbol M_4 in Fig. 3 indicates the entirety of the change portion, that is, start part+intermediate part+end part. At that time, the change portion of the presentation data D is replaced by a macro 'elem_users' designated by a symbol M_5. Further, 'id="users"' serving as a mark is removed.

[0032] Next, as shown in Fig. 4, presentation logic L is prepared as a macro.

[0033] This macro is prepared so as to overwrite the macro (here, 'elem_users') indicating the entirety of the change portion. In this example, by substituting each element of an array 'userlist' into each variable 'user' one by one, the macro 'body_users' serving as the intermediate part is repeated (that is, this repeating is the presentation logic L).

[0034] Then, as shown in Fig. 5, the above-described macro definitions are read, combined, and expanded.

[0035] First, since the change portion of the presentation data D has been replaced by the macro 'elem_users,' this is expanded (see the arrow X in Fig. 5).

[0036] Since the presentation logic L and the macro are

included in the expanded one, the macro is further expanded (see the arrow Y in Fig. 5).

[0037] As a result, a template T is obtained in which the presentation data D and the presentation logic L are combined (see Fig. 5).

[0038] The thus obtained template T is read by a main program, and then executed.

For example, when the contents of the array 'userlist' are ["foo," "bar," "baz"], an HTML file (a symbol F) as shown in Fig. 6 is generated.

[0039] In a technique in the general prior art, the above-described template T is needed to be directly generated. That is, the presentation data D and the presentation logic L have not been separated from each other, that is, have been integrated. In contrast, according to the present invention, it is recognized that the template T can be separated into the presentation data D (see Fig. 2) and the presentation logic L (see Fig. 4).

[0040] Here, Fig. 7 shows an example of presentation logic (see the symbol La in Fig. 7) in which the entirety including the start part and the end part is repeated.

[0041] Fig. 8 shows a modified embodiment of presentation logic (see the symbol Lb in Fig. 8) in which the

start part and the end part are kept from being outputted (that is, the tags are not outputted). When the tags are kept from being outputted as in this modified embodiment, output of a format other than HTML and XML can be obtained.

[0042] Fig. 9 shows a modified embodiment describing complicated presentation logic such as a conditional branch (see the symbol Lc in Fig. 9).

[0043] Other embodiments of the present invention are described below with reference mainly to Figs. 10 through 14. These embodiments employ a configuration in which abstracted elements serving as a start part, an intermediate part, and an end part in the presentation data D are divided into two abstracted elements composed of paired 'start part/end part' and an 'intermediate part.'

[0044] First, Fig. 10 shows a program for generating the same XML as the presentation data in an XML format shown in Fig. 2 (the tag with 'id="users"' is a change portion). In this program, a method 'addElementUsers()' indicates a start part and an end part, while a method 'addContentUsers()' indicates an intermediate part (however, 'id="users"' is assumed to be deleted in the method 'addElementUsers()'). A method generate() is a method for outputting the entirety of the presentation data. When this method is executed, an XML

equivalent to the presentation data of Fig. 2 is generated.

[0045] Further, such a program can be automatically generated from an XML serving as the presentation data. That is, the program of Fig. 10 can be considered as being equivalent to the presentation data of Fig. 2, although they are different in formats from each other.

[0046] Fig. 11 is a diagram showing a program that represents presentation logic. In this program, the program shown in Fig. 10 is inherited. Further, the method 'addContentUsers()' is overridden so that the method having the same name in the parent class is repeatedly called. Here, the method 'addContentUsers()' in the parent class indicates the intermediate part. As a result, the program shown in Fig. 11 implements presentation logic in which the intermediate part is solely repeated.

[0047] In this program of Fig. 11, the method 'generate()' of Fig. 10 is inherited from the parent class. Thus, also in the program of Fig. 11, the method 'generate()' can be executed, although the method is not defined. As a result, for example, when the contents of the variable userlist is an array ["foo," "bar," "baz"], output results similar to that of Fig. 6 described above are obtained.

[0048] Here, in the presentation logic shown in Fig. 11,

neither tag name nor attribute name appears. That is, the presentation logic of Fig. 11 never includes the presentation data in Fig. 2. Further, the presentation data of Fig. 2 never includes the presentation logic. This indicates that the presentation logic and the presentation data are separated from each other.

[0049] Next, Fig. 12 shows an example of presentation logic in which the entirety of the change portion including the start part and the end part is repeated. Fig. 13 shows an example of presentation logic in which the intermediate part is solely repeated without outputting the start part and the end part. Fig. 14 shows an example of complicated presentation logic including a repeat and a conditional branch.

[0050] As described above, also in the embodiment in which the change portion of the presentation data is divided into 'start part/end part' and an 'intermediate part,' the presentation logic and the presentation data can be separated from each other even by a method for operating these change portions arbitrarily from the presentation logic.

[0051] As seen from the above-described embodiments, in the present invention, even when the presentation logic L is complicatedly changed, the presentation data D need not be

changed at all. That is, it is recognized that the presentation data D and the presentation logic L are separated from each other.

[0052] Further, in the presentation logic L, since the change portion in the presentation data D is abstracted into start part/intermediate part/end part, an arbitrary operation is allowed such that the intermediate part is solely repeated or alternatively the entirety of start part/intermediate part/end part is repeated.

Industrial Applicability

[0053] The present invention permits output of data in an arbitrary format, and hence has a wide range of applications. Thus, in addition to Web applications for outputting an HTML and an XML, the present invention is widely applicable, for example, to the generation of various kinds of configuration files or communication data (for example, a data file or data such as an XML file for exchanging information).

BRIEF DESCRIPTION OF THE DRAWINGS

[0054]

[Fig. 1] is a diagram showing a basic concept of a template system according to the present invention;

[Fig. 2] is an example of presentation data (in an XML format) in which a mark is attached;

[Fig. 3] is an example of a macro automatically generated from presentation data;

[Fig. 4] is an example of presentation logic generated by overwriting a macro;

[Fig. 5] is an example of a template automatically generated by combining macros;

[Fig. 6] is a diagram showing an exemplary output in a case that the contents of a variable userlist are an array ["foo," "bar," "baz"];

[Fig. 7] is a diagram showing a modified embodiment of presentation logic (an example in which repeating is performed including tags);

[Fig. 8] is a diagram showing a modified embodiment of presentation logic (an example in which tags are not outputted);

[Fig. 9] is a diagram showing a modified embodiment of presentation logic (an example including a conditional branch);

[Fig. 10] shows a program for generating the same XML as presentation data in an XML format shown in Fig. 2 (the tag with 'id="users"' is a change portion);

[Fig. 11] is a diagram showing a program that represents presentation logic;

[Fig. 12] is a diagram showing an example of presentation logic in which the entirety of the change portion including a start part and an end part is repeated;

[Fig. 13] is a diagram showing an example of presentation logic in which an intermediate part is solely outputted without outputting a start part and an end part;

[Fig. 14] is a diagram showing an example of complicated presentation logic including a repeat and a conditional branch; and

[Fig. 15] is a diagram for describing a conventional template (a template in which presentation data and presentation logic are mixed).

Description of the Symbols

[0055]

D Presentation data

L, La, Lb, and Lc Presentation logic (separated from presentation data)

M_1 Macro indicating start part of change portion

M_2 Macro indicating intermediate part of change portion

M_3 Macro indicating end part of change portion

M_4 Macro indicating entirety of change portion

M_5 Macro for converting presentation data

P Presentation layer